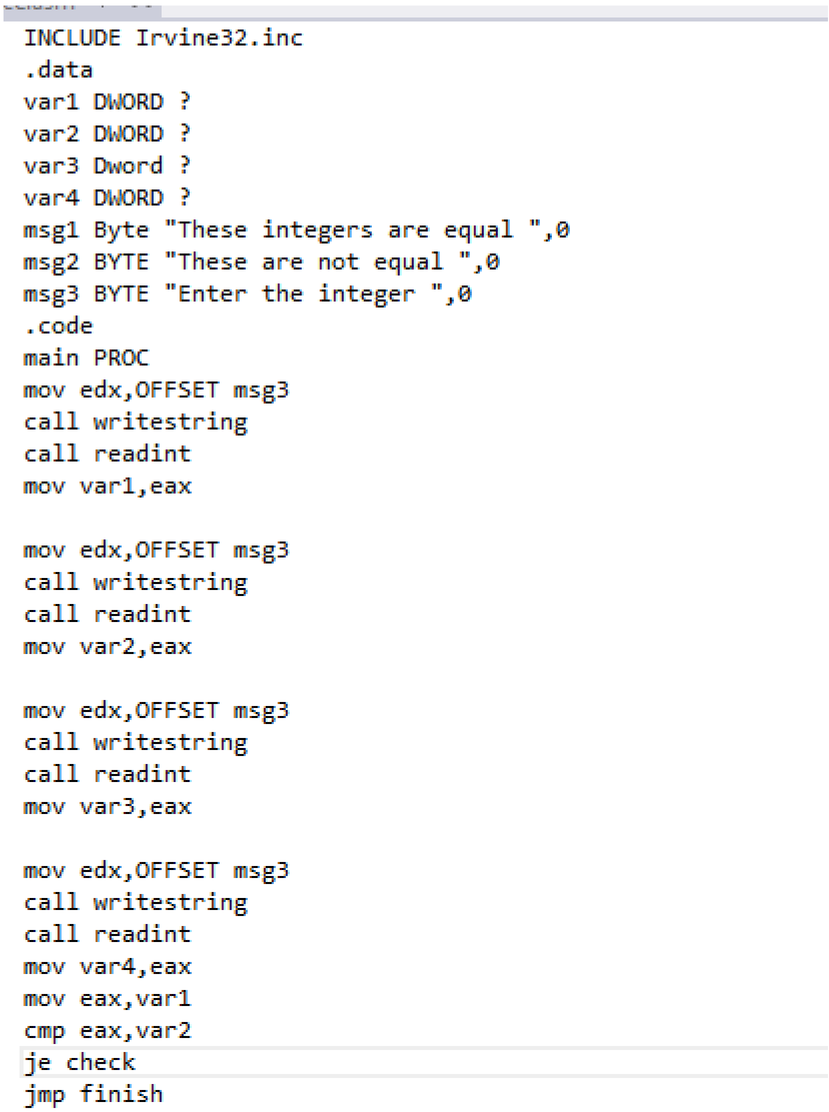
Muzamil

Lab#8

Task#1

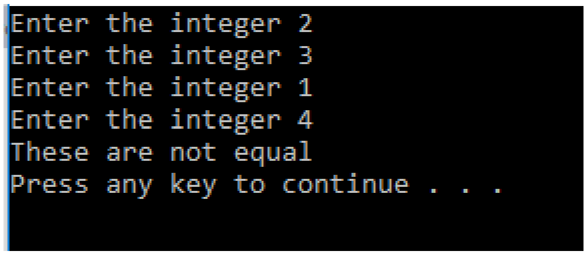


```
INCLUDE Irvine32.inc
.data
var1 DWORD ?
var2 DWORD ?
var3 Dword ?
var4 DWORD ?
msg1 Byte "These integers are equal ",0
msg2 BYTE "These are not equal ",0
msg3 BYTE "Enter the integer ",0
.code
main PROC
mov edx,OFFSET msg3
call writestring
call readint
mov var1,eax

mov edx,OFFSET msg3
call writestring
call readint
mov var2,eax

mov edx,OFFSET msg3
call writestring
call readint
mov var3,eax

mov edx,OFFSET msg3
call writestring
call readint
mov var4,eax
mov eax,var1
cmp eax,var2
je check
jmp finish
```

```
check:
mov eax,var2
cmp eax,var3
je next
jmp finish

next:
mov eax,var3
cmp eax,var4
je last
jmp finish
last:
mov edx,offset msg1
call writestring
call crlf
call khtm
finish:
mov edx,offset msg2
call writestring
call crlf

khtm:


exit
main ENDP
END main
```

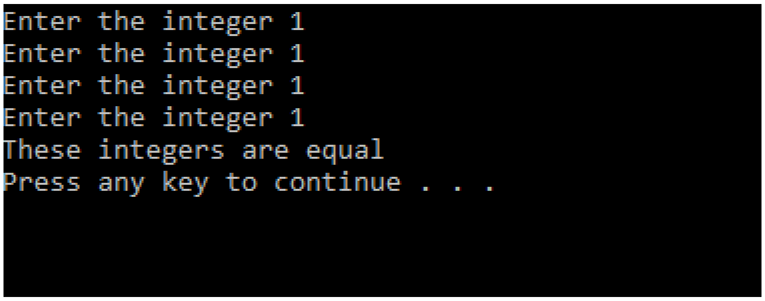
Output

Not equal



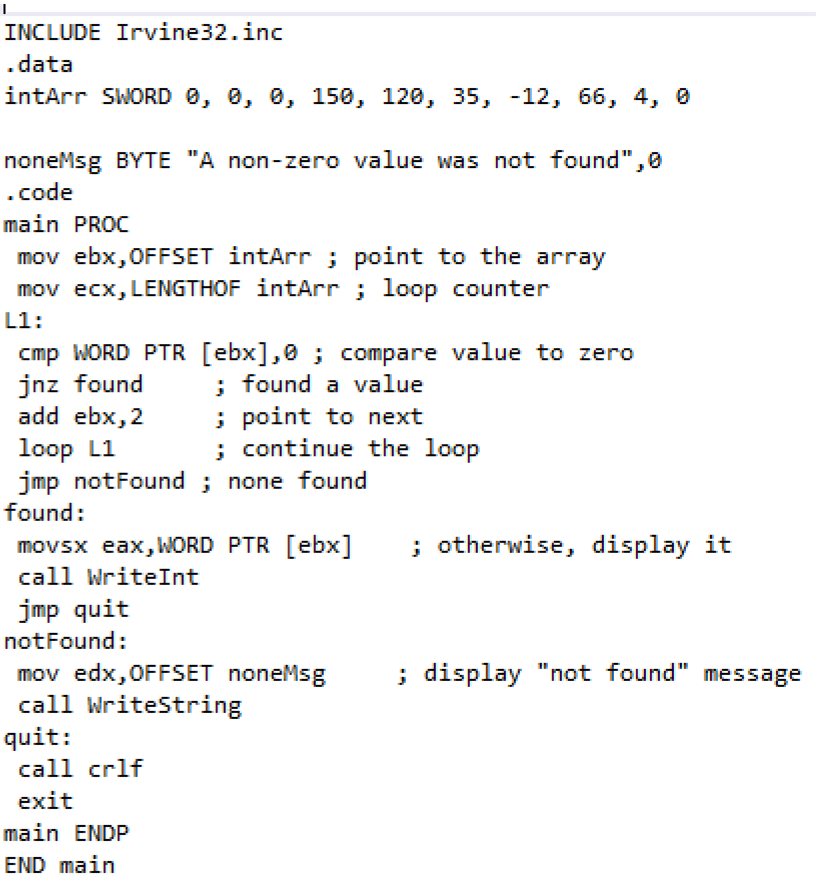
```
Enter the integer 2
Enter the integer 3
Enter the integer 1
Enter the integer 4
These are not equal
Press any key to continue . . .
```

Equal



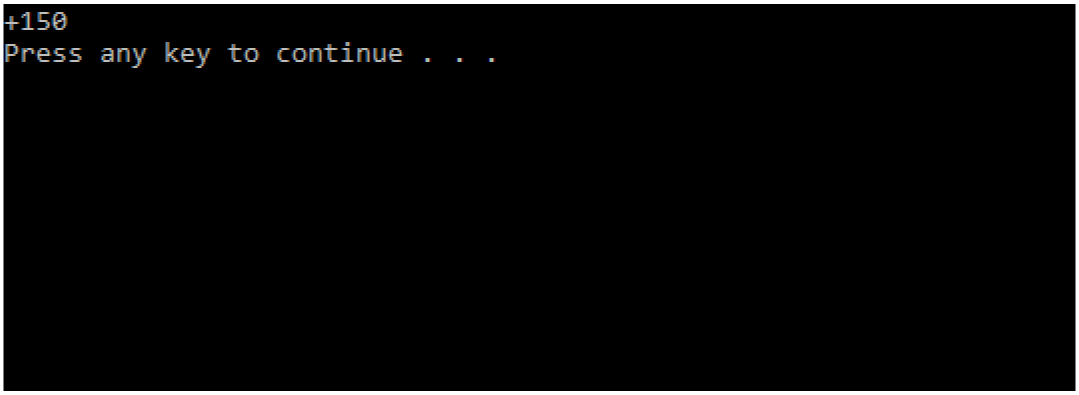
```
Enter the integer 1
Enter the integer 1
Enter the integer 1
Enter the integer 1
These integers are equal
Press any key to continue . . .
```

Task#2



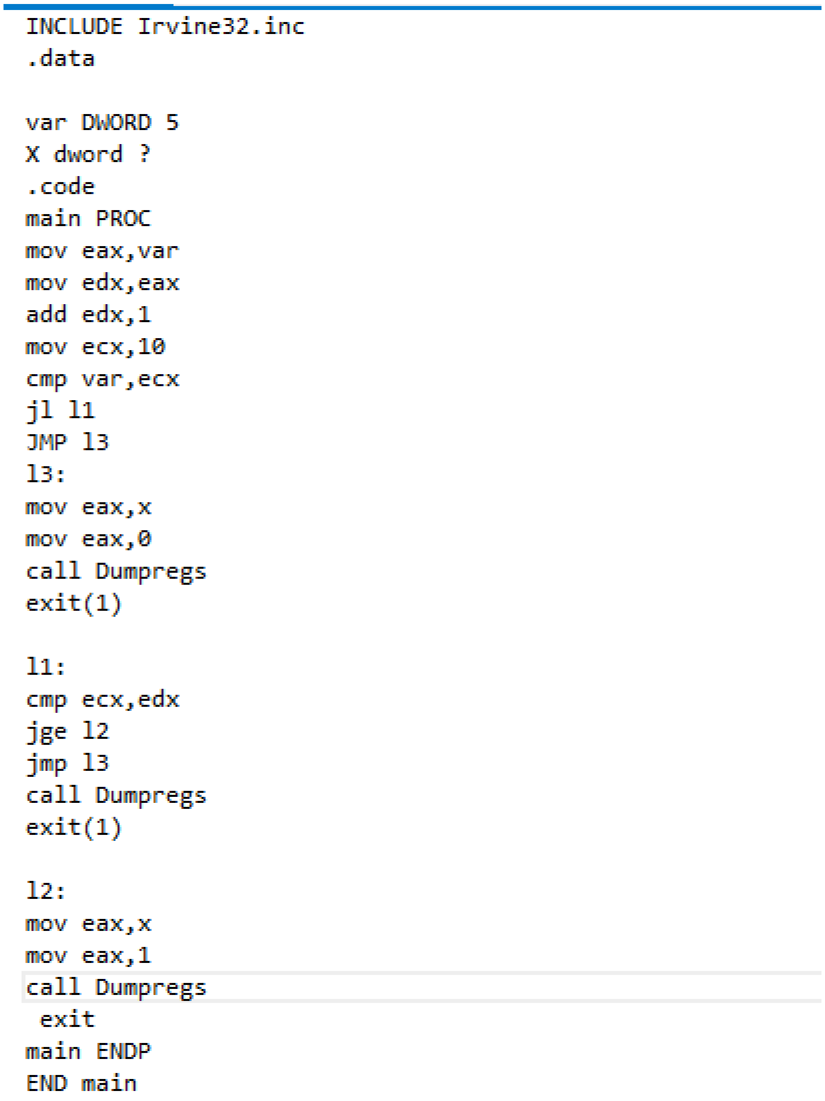
```
INCLUDE Irvine32.inc
.data
intArr SWORD 0, 0, 0, 150, 120, 35, -12, 66, 4, 0

noneMsg BYTE "A non-zero value was not found",0
.code
main PROC
 mov ebx,OFFSET intArr ; point to the array
 mov ecx,LENGTHOF intArr ; loop counter
L1:
 cmp WORD PTR [ebx],0 ; compare value to zero
 jnz found      ; found a value
 add ebx,2      ; point to next
 loop L1        ; continue the loop
 jmp notFound ; none found
found:
 movsx eax,WORD PTR [ebx]    ; otherwise, display it
 call WriteInt
 jmp quit
notFound:
 mov edx,OFFSET noneMsg     ; display "not found" message
 call WriteString
quit:
 call crlf
 exit
main ENDP
END main
```

Output



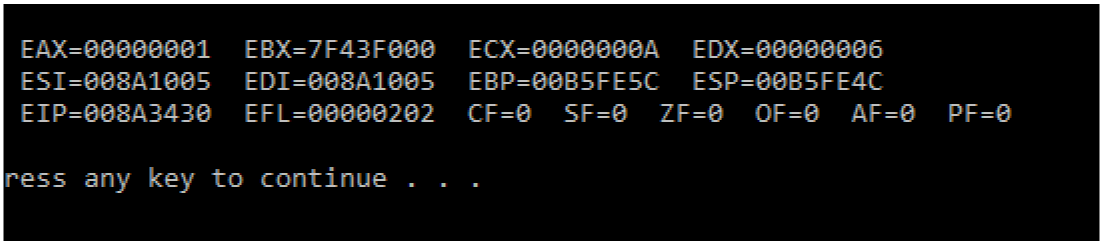
```
+150
Press any key to continue . . .
```

Task#3



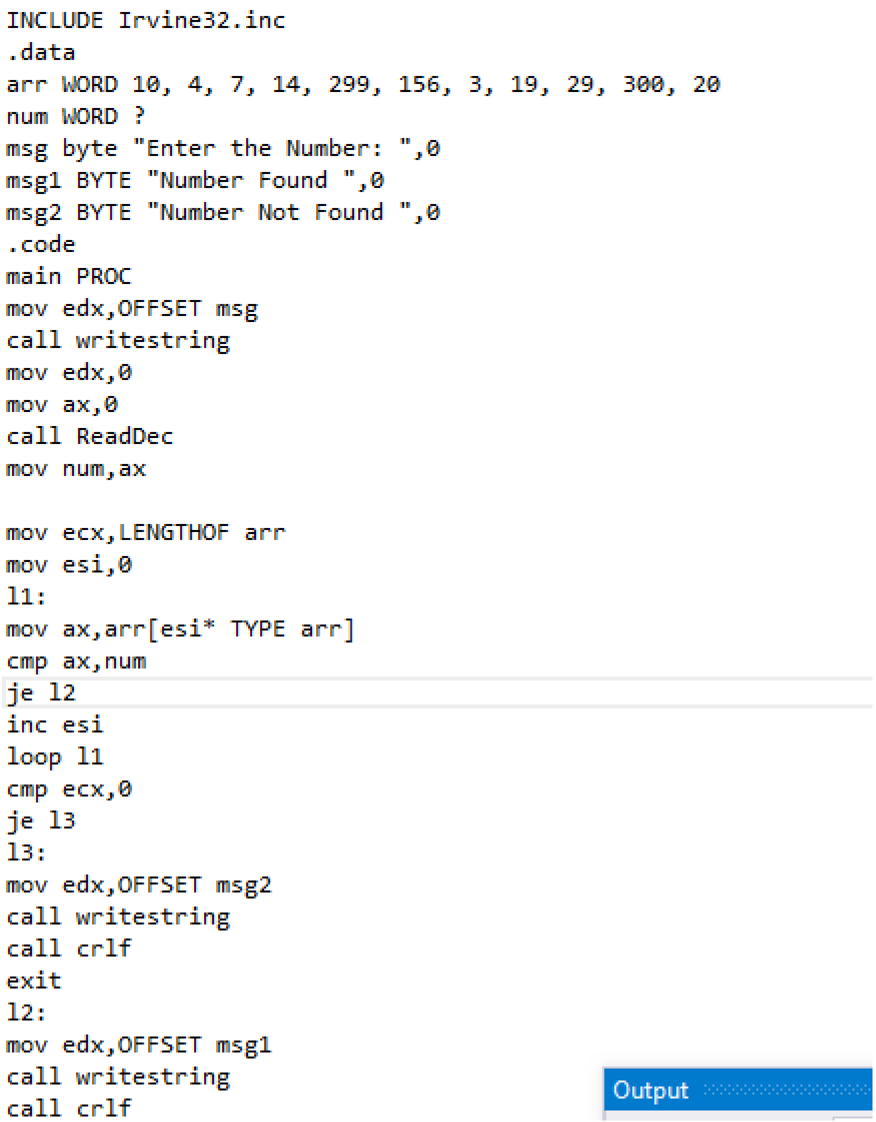
```
INCLUDE Irvine32.inc
.data

var DWORD 5
X dword ?
.code
main PROC
mov eax,var
mov edx,eax
add edx,1
mov ecx,10
cmp var,ecx
jl l1
JMP l3
l3:
mov eax,x
mov eax,0
call Dumpregs
exit(1)

l1:
cmp ecx,edx
jge l2
jmp l3
call Dumpregs
exit(1)

l2:
mov eax,x
mov eax,1
call Dumpregs
 exit
main ENDP
END main
```

Output



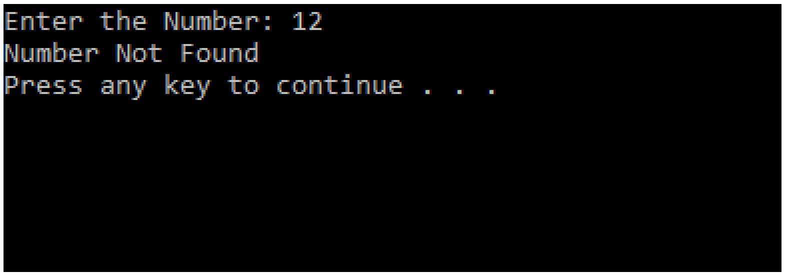
```
 EAX=00000001  EBX=7F43F000  ECX=0000000A  EDX=00000006
 ESI=008A1005  EDI=008A1005  EBP=00B5FE5C  ESP=00B5FE4C
 EIP=008A3430  EFL=00000202  CF=0  SF=0  ZF=0  OF=0  AF=0  PF=0

ress any key to continue . . .
```

Task#5

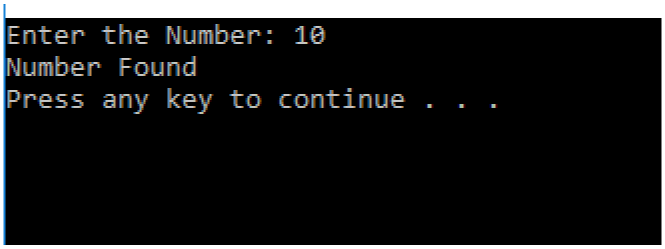


```
INCLUDE Irvine32.inc
.data
arr WORD 10, 4, 7, 14, 299, 156, 3, 19, 29, 300, 20
num WORD ?
msg byte "Enter the Number: ",0
msg1 BYTE "Number Found ",0
msg2 BYTE "Number Not Found ",0
.code
main PROC
mov edx,OFFSET msg
call writestring
mov edx,0
mov ax,0
call ReadDec
mov num,ax

mov ecx,LENGTHOF arr
mov esi,0
l1:
mov ax,arr[esi* TYPE arr]
cmp ax,num
je l2
inc esi
loop l1
cmp ecx,0
je l3
l3:
mov edx,OFFSET msg2
call writestring
call crlf
exit
l2:
mov edx,OFFSET msg1
call writestring
call crlf
```

Output

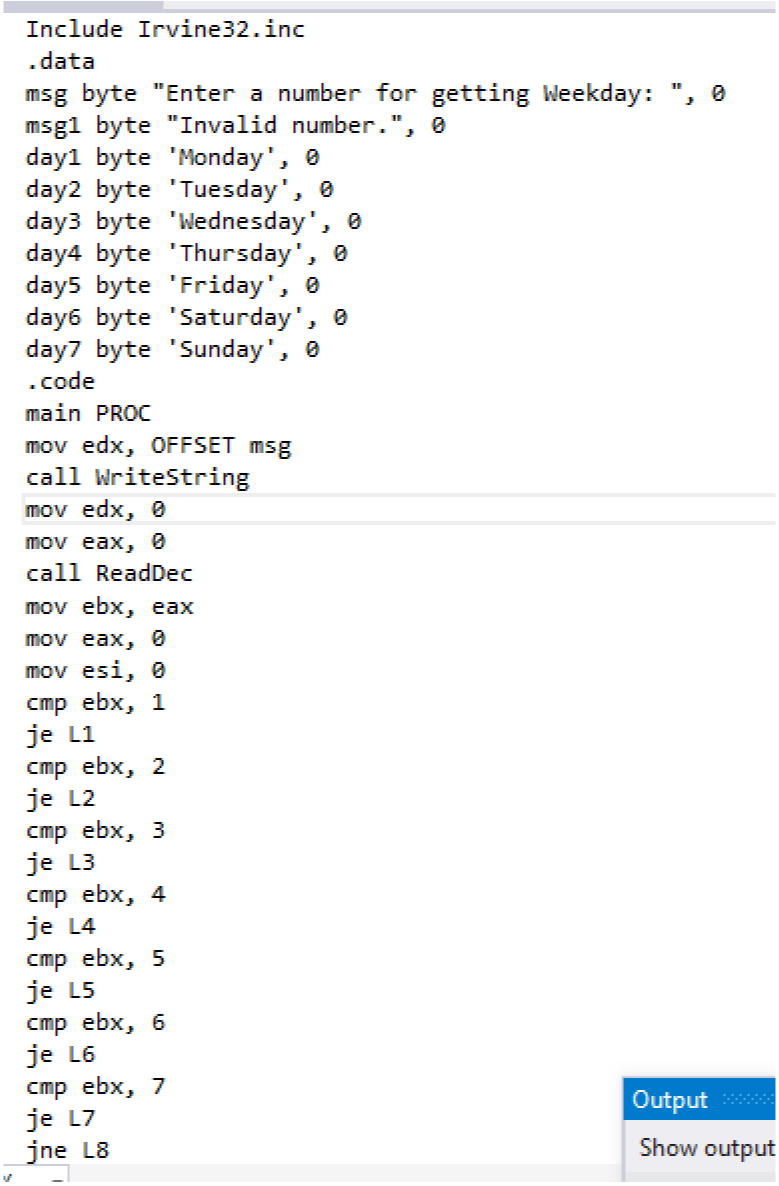


```
Enter the Number: 12
Number Not Found
Press any key to continue . . .
```

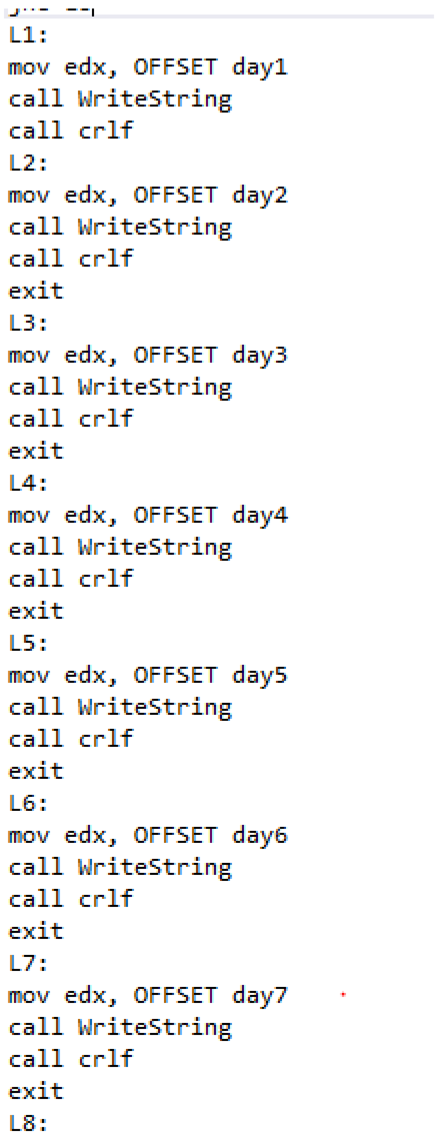


```
Enter the Number: 10
Number Found
Press any key to continue . . .
```

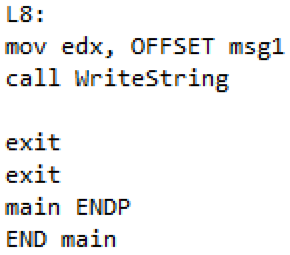
Task#7



```
Include Irvine32.inc
.data
msg byte "Enter a number for getting Weekday: ", 0
msg1 byte "Invalid number.", 0
day1 byte 'Monday', 0
day2 byte 'Tuesday', 0
day3 byte 'Wednesday', 0
day4 byte 'Thursday', 0
day5 byte 'Friday', 0
day6 byte 'Saturday', 0
day7 byte 'Sunday', 0
.code
main PROC
mov edx, OFFSET msg
call WriteString
mov edx, 0
mov eax, 0
call ReadDec
mov ebx, eax
mov eax, 0
mov esi, 0
cmp ebx, 1
je L1
cmp ebx, 2
je L2
cmp ebx, 3
je L3
cmp ebx, 4
je L4
cmp ebx, 5
je L5
cmp ebx, 6
je L6
cmp ebx, 7
je L7
jne L8
```

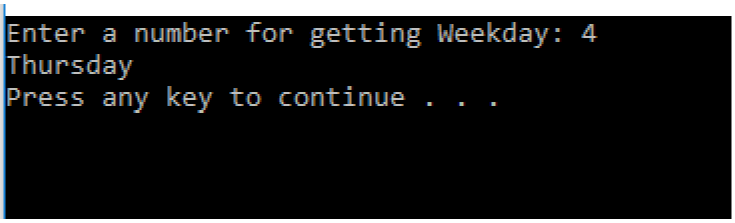


```
L1:
mov edx, OFFSET day1
call WriteString
call crlf
L2:
mov edx, OFFSET day2
call WriteString
call crlf
exit
L3:
mov edx, OFFSET day3
call WriteString
call crlf
exit
L4:
mov edx, OFFSET day4
call WriteString
call crlf
exit
L5:
mov edx, OFFSET day5
call WriteString
call crlf
exit
L6:
mov edx, OFFSET day6
call WriteString
call crlf
exit
L7:
mov edx, OFFSET day7
call WriteString
call crlf
exit
L8:
```



```
L8:
mov edx, OFFSET msg1
call WriteString

exit
exit
main ENDP
END main
```

Output



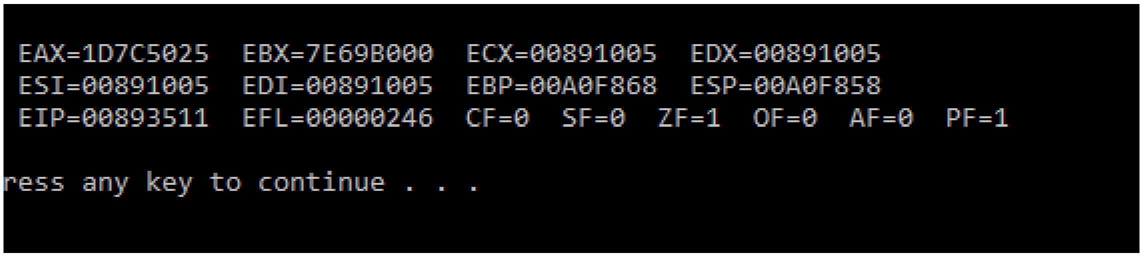
```
Enter a number for getting Weekday: 4
Thursday
Press any key to continue . . .
```

Example#1

```
Include Irvine32.inc
.code
main PROC
mov al, 10101110b ; Clear only bit 3
and al, 11110110b ; AL = 10100110
mov al, 11100011b ; set bit 2
or al, 00000100b ; AL = 11100111
mov al, 10110101b ; 5 bits means odd parity
xor al, 0 ;PF=0(PO)
mov al, 10100101b ; 4 bits means even parity
xor al, 0 ;PF=1(PE)
```

```
mov al, 11110000b
not al ; AL = 00001111b
mov al, 00100101b
test al, 00001001b ;ZF=0
mov al, 00100101b
test al, 00001000b ;ZF=1
call DumpRegs


exit
main ENDP
END main
```

Output

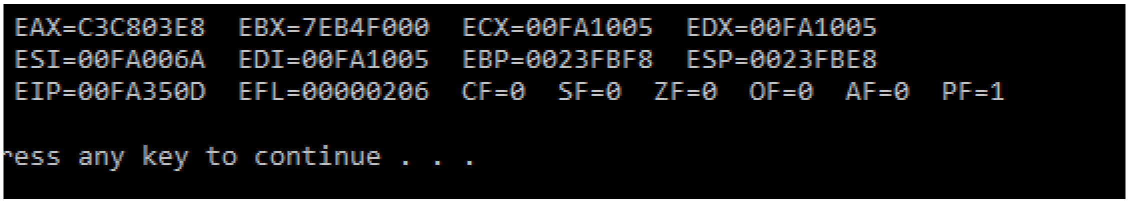


```
 EAX=1D7C5025  EBX=7E69B000  ECX=00891005  EDX=00891005
 ESI=00891005  EDI=00891005  EBP=00A0F868  ESP=00A0F858
 EIP=00893511  EFL=00000246  CF=0  SF=0  ZF=1  OF=0  AF=0  PF=1

ress any key to continue . . .
```

Example#2

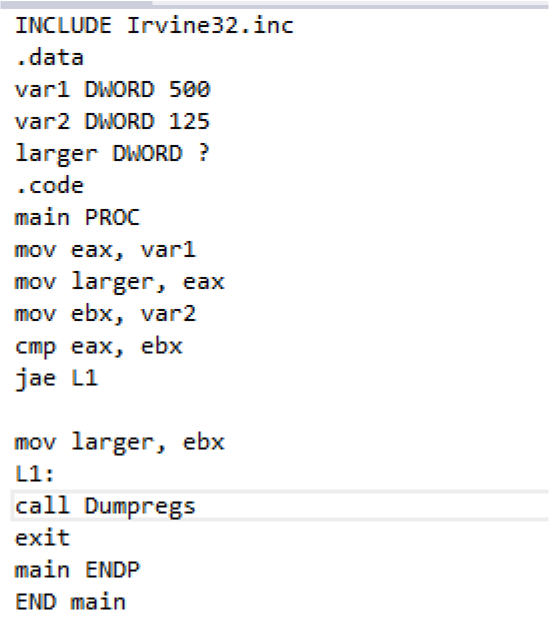
```
Include Irvine32.inc
.code
main PROC

mov ax, 5
cmp ax, 10 ; ZF = 0 and CF = 1


mov ax, 1000
cmp ax, 1000 ;ZF=1 and CF =0
mov si, 106
cmp si, 0 ;ZF=0 and CF=0
call DumpRegs


exit
main ENDP
END main
```

Output



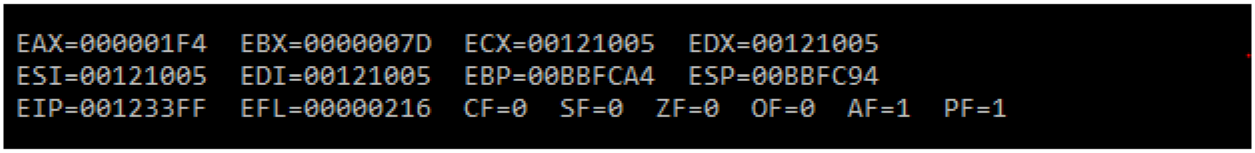
```
EAX=C3C803E8  EBX=7EB4F000  ECX=00FA1005  EDX=00FA1005
ESI=00FA006A  EDI=00FA1005  EBP=0023FBF8  ESP=0023FBE8
EIP=00FA350D  EFL=00000206  CF=0  SF=0  ZF=0  OF=0  AF=0  PF=1

ress any key to continue . . .
```

Example#3



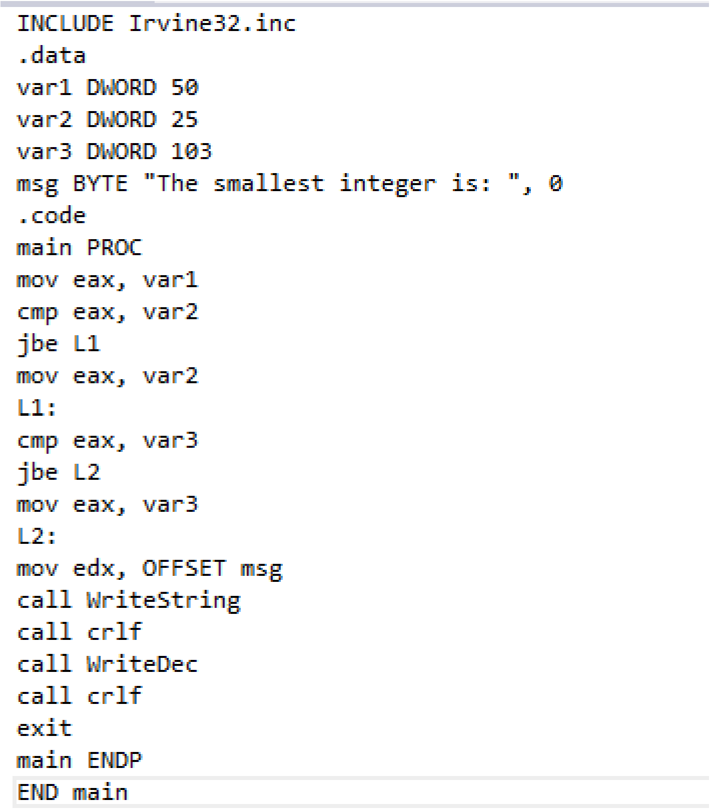
```
INCLUDE Irvine32.inc
.data
var1 DWORD 500
var2 DWORD 125
larger DWORD ?
.code
main PROC
mov eax, var1
mov larger, eax
mov ebx, var2
cmp eax, ebx
jae L1

mov larger, ebx
L1:
call Dumpregs
exit
main ENDP
END main
```

OutPut



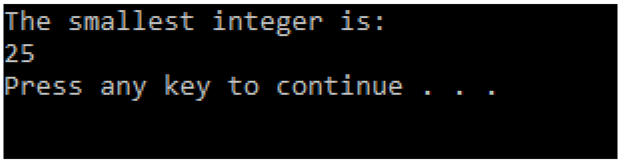
```
EAX=000001F4  EBX=0000007D  ECX=00121005  EDX=00121005
ESI=00121005  EDI=00121005  EBP=00BBFCA4  ESP=00BBFC94
EIP=001233FF  EFL=00000216  CF=0  SF=0  ZF=0  OF=0  AF=1  PF=1
```

Example#4



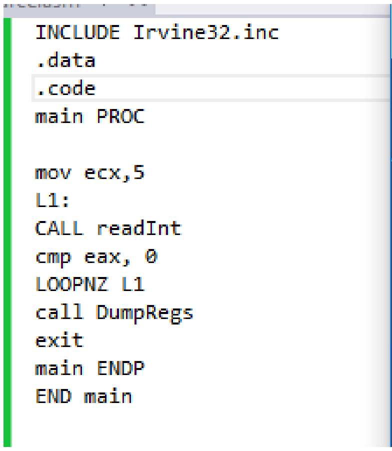
```
INCLUDE Irvine32.inc
.data
var1 DWORD 50
var2 DWORD 25
var3 DWORD 103
msg BYTE "The smallest integer is: ", 0
.code
main PROC
mov eax, var1
cmp eax, var2
jbe L1
mov eax, var2
L1:
cmp eax, var3
jbe L2
mov eax, var3
L2:
mov edx, OFFSET msg
call WriteString
call crlf
call WriteDec
call crlf
exit
main ENDP
END main
```

Output

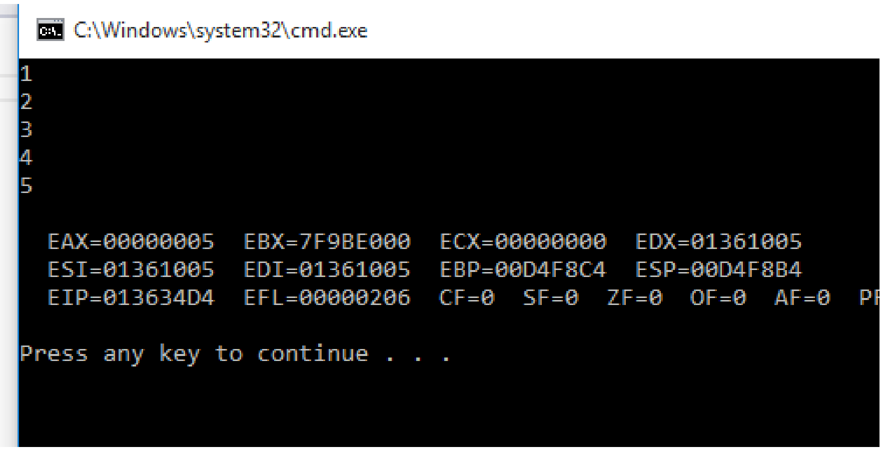


```
The smallest integer is:
25
Press any key to continue . . .
```

Example#5 &Itz Output



```
INCLUDE Irvine32.inc
.data
.code
main PROC

mov ecx,5
L1:
CALL readInt
cmp eax, 0
LOOPNZ L1
call DumpRegs
exit
main ENDP
END main
```



```
1
2
3
4
5

  EAX=00000005  EBX=7F9BE000  ECX=00000000  EDX=01361005
  ESI=01361005  EDI=01361005  EBP=00D4F8C4  ESP=00D4F8B4
  EIP=013634D4  EFL=00000206  CF=0  SF=0  ZF=0  OF=0  AF=0  PF

Press any key to continue . . .
```